DVD NAVIGATOR AND APPLICATION PROGRAMMING INTERFACES

Publication number: KR20020040588 (A)
Publication date: 2002-05-30
Inventor(s): CHAKRABARTI ALOK; EVANS GLENN F
Applicant(s): MICROSOFT CORP
Classification:
- international: ***G11B20/10; G06F9/46; G11B19/02; G11B27/10; H04N5/85;*** **G11B20/10; G06F9/46; G11B19/02; G11B27/10; H04N5/84;** (IPC1-7): G11B19/02
- European: G06F9/46R6B; G11B19/02; G11B27/10A1; H04N5/85
Application number: KR20010072628 20011121
Priority number(s): US20000721402 20001122

Also published as:
EP1251514 (A2)
EP1251514 (A3)
US7451453 (B1)
KR20080103041 (A)
JP2002288934 (A)

Abstract of **KR 20020040588 (A)**

PURPOSE: DVD navigator and application programming interfaces are provided to extend the performance of a generic DVD navigator component and obtain better control mechanisms for player applications. CONSTITUTION: DVD navigator(106) and application programming interfaces comprise memory and logic operatively coupled to the memory and operatively configurable to access multimedia content from a medium. The logic provides a multimedia navigator program, a control application programming interface(API) and an information API. The control and information APIs are configured to respond to flags selectively determining if at least one operation is conducted. The operation is selected from a group of operations including a player-navigator synchronization operation, a selective interactive user operation, and a read/write register operation.

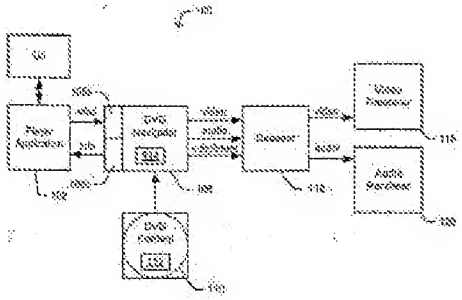


Fig. 1

(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(51) 。Int. Cl. [7]
G11B 19/02

(11) 공개번호 특2002- 0040588
(43) 공개일자 2002년05월30일

(21) 출원번호 10- 2001- 0072628
(22) 출원일자 2001년11월21일

(30) 우선권주장 09/721,402 2000년11월22일 미국(US)

(71) 출원인 마이크로소프트 코포레이션
존 비. 메이슨
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이

(72) 발명자 이반스글렌에프.
미국98034워싱턴주컬크랜드엔이원헌드레드써티써드피엘.7833
챠크라바르티알룩
미국98006워싱턴주벨리브원헌드레드포티퍼스트피엘에스이5724

(74) 대리인 주성민
백만기
이중희

심사청구 : 없음

(54) 향상된 D V D 네비게이터 및 애플리케이션 프로그래밍인터페이스

요약

본 발명의 특성에 따르면, 포괄 DVD 네비게이터 컴포넌트의 성능을 더 확장시키기 위해 확장 소자들이 개발되었다. 여기서, DVD 네비게이터 프로그램은 DVD 플레이어 애플리케이션들과 더 양호하게 동기화하도록 확장되었다. 따라서, 플레이어 애플리케이션을 위한 양호한 제어 메카니즘들이 제공된다. 예를 들어, 한정/부모 제어 콘텐트 뷰의 향상된 수행, 콘텐트 소스에 대한 고유 식별자 생성 및 할당과, 재생 시작 및 정지를 위한 향상된 사용자 및 플레이어 애플리케이션 환경이 제공된다.

대표도
도 1

색인어
플레이어 애플리케이션, DVD 네비게이터, DVD 콘텐트, 사용자 인터페이스, 애플리케이션 프로그래밍 인터페이스(API)

명세서

도면의 간단한 설명

도 1은 DVD 플레이어 장치의 일례를 도시하는 블럭도.

도 2는 도 1의 DVD 플레이어 장치와 함께 사용되기에 적합한 컴퓨터 환경을 도시하는 블럭도.

도 3은 DVD 플레이어 애플리케이션과 포괄 네비게이터 프로그램 간의 제1 동기화 모드를 도시하는 블럭도.

도 4는 DVD 플레이어 애플리케이션과 포괄 네비게이터 프로그램 간의 제2 동기화 모드를 도시하는 블럭도.

도 5는 DVD 플레이어 애플리케이션과 포괄 네비게이터 프로그램 간의 제3 동기화 모드를 도시하는 블럭도.

도 6은 DVD 플레이어 애플리케이션과 포괄 네비게이터 프로그램 간의 제4 동기화 모드를 도시하는 블럭도.

도 7 및 도 8은 DVD 플레이어 애플리케이션과 포괄 네비게이터 프로그램 간의 논- 블럭킹 동기화 모드 및 블럭킹 동기화 모드를 각각 도시하는 블럭도.

도 9는 플레이어 애플리케이션과 미디어 콘텐트 관련 프로그램 간의 판독/기록 통신 기능의 일례를 도시하는 블럭도.

도 10은 미디어 콘텐트에 대한 한정/부모 제어와 관련된 이중- 분기(dual- branch) 재생 결정 지점을 도시하는 배선도.

도 11은 미디어 콘텐트에 대한 한정/부모 제어와 관련된 다중- 분기 재생 결정 지점을 도시하는 배선도.

도 12는 플레이어 애플리케이션 제공 코드를 사용해서 미디어 데이터에 대한 액세스를 제어하기 위한 방법의 일례를 도시하는 블럭도.

도 13은 미디어 콘텐트 북마킹 기능의 일례를 도시하는 블럭도.

도 14는 미디어 소스를 위한 실제 고유 식별자를 생성하는 방법의 일례를 도시하는 도면.

< 도면의 주요 부분에 대한 부호의 설명>

100 : DVD 플레이어

102 : 플레이어 애플리케이션

104 : 사용자 인터페이스

106 : 네비게이터

108a- b : DVD2 API

110 : DVD 콘텐트

112 : 프로그램

200 : 컴퓨팅 시스템

222 : 시스템 메모리

223 : 시스템 버스

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 특허 출원은 함께 계류 중인 미국 특허 번호들 제_____호 (위임장 문서 번호: MSI- 703US, MSI- 706US, MSI- 707US, MSI- 708US, 및 MSI- 709US)와 관련된다.

본 발명은 컴퓨터 등의 장치들에 관한 것으로, 특히 포괄(generic) 네비게이터 프로그램 및 공개된 애플리케이션 프로그래밍 인터페이스(application programming interface: API)와 연관된 향상된 방법들 및 장치들에 관한 것이다.

DVD(digital versatile disc) 플레이어는 DVD 명세서에 정의된 바와 같이 3개의 논리 유닛들로 구성된다. 제1 논리 유닛은 사용자에게 인터페이스를 제공하고 사용자 커맨드를 제2 논리 유닛과 연관시키는 DVD 플레이어 애플리케이션이다. 제2 논리 유닛은 DVD에 대한 정보를 판독 및 해석하고 사용자 커맨드를 기초로 비디오 및 오디오의 어떤 세그먼트들이 처리될 것인지를 제어하는 DVD 네비게이터이다. 제3 논리 유닛은 DVD로부터 판독된 데이터를 압축 해제하고 응용 가능한 대응 오디오, 비디오 및 서브픽쳐 스트림들을 하나 이상의 렌더들에게 제공하는 DVD 프리젠테이션층이다.

상기 논리 유닛들은 하드웨어 및/또는 소프트웨어로 구현될 수 있다. 예를 들어, 특정 구현에서, DVD 플레이어는 사용자에게 디스플레이되는 그래픽 사용자 인터페이스(graphic user interface: GUI)를 통해 조작되는데, 사용자는 GUI를 통해 예를 들면 마우스와 같은 포인팅 선택 입력 장치를 사용해서 DVD의 재생 등을 선택적으로 제어할 수 있다. 이는 대체로 시스템 개발자에게는 꽤 간단한 작업이며 커스터마이징을 용이하게 할 수 있게 한다.

한편, DVD 네비게이터를 구현하는 것은 보다 복잡한 작업이 될 수 있다. 이는 특히 DVD 정보를 프리젠테이션 등으로 통합하고자 하는 애플리케이션의 경우에 더 그러하다. 여기서, 각각의 개발자 엔티티(entity)는 DVD를 판독 및 해석하고, DVD 프리젠테이션층의 디코더 메카니즘들과 인터페이스하기 위한 메카니즘을 제공할 필요가 있다. 또한, DVD 프리젠테이션층의 디코더 메카니즘은 제3자의 산출물일 수 있는데; DVD 네비게이터가 다수의 상이한 디코더 메카니즘들과 인터페이스해야만 하기 때문에, DVD 네비게이터를 저작하는 작업을 훨씬 더 어렵게 할 가능성이 있다.

따라서, 플레이어 애플리케이션들이 DVD 네비게이터 프로그램을 제어하기 위해 사용할 수 있는 강력하면서도 간단하고 일관성 있는 인터페이스가 필요하다.

발명이 이루고자 하는 기술적 과제

오퍼레이팅 시스템 및 사용자 환경을 더 강화하고자 노력하는 마이크로소프트 주식회사 애플리케이션 개발자들에게 부과된 잠정적인 부담을 인식해서 포괄 네비게이터 컴포넌트(generic DVD navigator component)를 개발해 왔다. 포괄 네비게이터 컴포넌트는 애플리케이션 개발자들이 가능성 있는 반복적이고 어려운 작업들을 피할 수 있도록 돕기 위해 윈도우즈의 일부로서 표준적이고 명세에 따르는 DVD 네비게이터를 제공한다. 이러한 포괄 네비게이터 컴포넌트는 DVD 네비게이터를 제어하기 위해 플레이어 애플리케이션들이 사용할 수 있는 강력하면서도 간단하고 일관성 있는 인터페이스를 제공하는 2개의 결합된 애플리케이션 프로그래밍 인터페이스(API)들을 공개한다. API들은 기본 DVD 네비게이터의 유연성과 유용성에 더 영향을 주도록 설계되었다.

본 발명의 특성들에 따르면, 포괄 네비게이터 컴포넌트의 성능을 더 확장시키기 위해 확장 소자(enhancement)들이 개발되었다. 중요한 점은 포괄 네비게이터 프로그램과 함께 사용될 향상된 API들이 필요하다는 점이었다. 여기서, 플레이어 애플리케이션과 더 양호하게 동기화하도록 네비게이터 프로그램이 확장되었고, 그렇게 함으로써, 플레이어 애플리케이션을 위한 더 양호한 제어 메카니즘들, 한정/부모(restricted/parental) 제어 콘텐트(content) 뷰의 향상된 시행, 콘텐트 소스에 대한 고유 식별자들의 생성 및 할당과, 재생 시작 및 정지를 위한 향상된 사용자 및 플레이어 애플리케이션 환경이 제공된다.

발명의 구성 및 작용

이하의 예시가 되는 방법과 장치들은 DVD 플레이어 애플리케이션과 관련된 API들을 갖는 포괄 DVD 네비게이터와 연관된 특정 확장 소자들 및 기능들에 대해 기술되어 있다. 이들은 DVD 네비게이터 및 DVD2 API라고 지칭된다. 대부분의 설명이 윈도우즈 오퍼레이팅 시스템을 운영하는 PC와 관련된 것이지만, 다양한 방법들 및 장치들이 다른 오퍼레이팅 시스템들, 장치들 등에도 명백하게 응용될 수 있음을 주지하자. 또한, DVD라는 용어를 사용하는 것은 다른 미디어 포맷들을 제외하는 것을 의미하는 것이 아니다. 따라서, DVD 콘텐트 자체는 하드 드라이브, 콤팩트 디스크, 네트워크 등으로부터 유래된 것일 수도 있다.

후술되는 바와 같이, DVD 네비게이터 및/또는 DVD2 API는 플레이어 애플리케이션이 DVD 콘텐트의 재생을 상호적으로 제어할 수 있게 해 준다. DVD2 API는 2개의 인터페이스들로 구성된다. 제1 인터페이스는 " IDvdInfo2" 라고 한다. 제2 인터페이스는 " IDvdControl2" 라고 한다. 플레이어 애플리케이션은 DVD 네비게이터의 현재 상태를 문의하기 위해 IDvdInfo2 인터페이스를 사용하고 재생을 양호하게 제어하고 및/또는 DVD 네비게이터의 상태를 변경하기 위해 IDvdControl2 인터페이스를 사용할 수도 있다.

DVD2 API는 몇몇 고유하고 신규한 기능들을 제공한다. 예를 들어, 실시간 재생을 위해 스레드- 베이스 동기화 방법들이 제공되고; 상호 작용성(interactivity)의 정도를 결정하기 위해 재생 제어 메카니즘이 제공되며; 플레이어 애플리케이션과 디스크 프로그램 간의 통신 메카니즘들이 제공되고, 시간 범위 플레이(playing of time range)가 지원되며; 부모 레벨 요청들을 조정하고 처리하며 콘텐트의 한정된 세그먼트를 플레이하도록 최소 부모 레벨을 결정하기 위한 메카니즘들이 제공되고; 고유 디스크 식별자 알고리즘이 제공되어, DVD 콘텐트 내의 임의의 위치의 북마킹을 더 지원한다.

상술된 바를 염두에 두고, 도 1이 도시되었다. 도 1은 예시가 되는 DVD 플레이어(100)를 도시한다. 이 플레이어(100)는 사용자에게 사용자 인터페이스(user interface: U/I)(104)를 제공하도록 구성된 적어도 하나의 플레이어 애플리케이션(102)을 포함한다. U/I(104)를 통해, 사용자는 DVD 콘텐트(110)의 재생과 관련해서 플레이어 애플리케이션(102)에게 명령할 수 있다.

도시된 바와 같이, 플레이어 애플리케이션(102)에는 사용자 요청들을 전달하고 피드백 정보를 수신하기 위해 DVD2 API(108a 및 108b)가 각각 제공된다. DVD2 API(108a- b)는 네비게이터(106) 내의 기능들에 대한 액세스를 제공한다. 네비게이터(106)는 미디어 정보 외에도 프로그램(112)을 포함하는 DVD 콘텐트(110)와 상호 작용한다. 프로그램(112)은 나머지 콘텐트와 관련된 메뉴들, 점프들 등을 정의한다. 네비게이터(106)는 재생 프로세스와 관련된 상태(state)(114)를 포함한다. 상태(114)에는, 예를 들어 현재 사용자 오퍼레이션(user operation: UOP) (예를 들면, 플레이, 정지, 일시 중지, 리버스(reverse), 빨리 감기(fast- forward), 슬로 모션, 앵글(angle) 등)이 DVD 콘텐트

내의 현재 위치 (예를 들어, 챕터, 시간, 프레임) 및 최근 점프/UOP를 기록할 수 있는 특정 다른 레지스터들과 함께 저장된다.

네비게이터(106) 의 출력은 응용될 수 있는 인코드 비디오 스트림, 인코드 오디오 스트림, 및 서브픽쳐 스트림을 포함한다. 상기 출력들은 인코드 데이터를 디코드 (해독 및 압축 해제) 하고 애플리케이션 가능 비디오 렌더러(118) 또는 오디오 렌더러(120) 에게 대응 스트림들을 출력하도록 구성된 디코더(116) 에 입력된다. 렌더러(118) 는 예를 들어 비디오 모니터를 통해 비디오 정보가 사용자에게 디스플레이되게 한다. 렌더러(120) 는 예를 들어 하나 이상의 스피커들을 통해 오디오 정보가 청취자를 위해 재생되게 한다.

이제 도 2를 참조하면, 도 2는 도 1의 장치와 함께 사용되기에 적합한, 일례가 되는 컴퓨팅 시스템(200) 의 블럭도를 도시한다.

본 예에서, 컴퓨팅 시스템(200) 은 퍼스널 컴퓨터(PC) 형태이지만, 다른 예들에서, 컴퓨팅 시스템은 전용 서버(들), 특정 목적 장치, 어플라이언스(appliance), 핸드헬드 컴퓨팅 디바이스, 이동 전화 장치, 페이저 장치 등의 형태일 수도 있다.

도시된 바와 같이, 컴퓨팅 시스템(200) 은 프로세싱 유닛(221), 시스템 메모리(222), 및 시스템 버스(223) 를 포함한다. 시스템 버스(223) 는 시스템 메모리(222) 및 프로세싱 유닛(221) 을 포함하는 다양한 시스템 컴포넌트들을 함께 링크한다. 시스템 버스(223) 는 메모리 버스 또는 메모리 컨트롤러, 주변 버스, 및 다양한 버스 아키텍쳐들 중 임의의 아키텍쳐를 사용하는 로컬 버스를 포함하는 몇몇 타입의 버스 구조들 중 임의의 한 타입일 수 있다. 시스템 메모리(222) 는 전형적으로 판독 전용 메모리(read only memroy: ROM) (224) 및 랜덤 액세스 메모리(random access memory: RAM) (225) 를 포함한다. 예를 들어 작동 개시 중에 컴퓨팅 시스템(200) 내의 소자들 간의 정보 전달을 돕는 기본 루틴을 포함하는 기본 입출력 시스템(226) (BIOS) 은 ROM(224) 에 기억된다. 컴퓨팅 시스템(200) 은 하드 디스크 판독 및 기록을 위한 하드 디스크 드라이브(227), 제거 가능 자기 디스크(229) 판독 및 기록을 위한 자기 디스크 드라이브(228), 및 CD ROM 또는 다른 광 미디어와 같은 제거 가능 광 디스크(231) 의 판독 및 기록을 위한 광 디스크 드라이브(230) 를 더 포함한다. 하드 디스크 드라이브(227), 자기 디스크 드라이브(228) 및 광 디스크 드라이브(230) 는 각각 하드 디스크 드라이브 인터페이스(232), 자기 디스크 드라이브 인터페이스(233) 및 광 드라이브 인터페이스(234) 에 의해 시스템 버스(223) 에 연결된다. 상기 드라이브들 및 관련 컴퓨터- 판독 가능 미디어는 컴퓨팅 시스템(200) 을 위해 컴퓨터 판독 가능 명령들, 데이터 구조들, 컴퓨터 프로그램들 및 다른 데이터의 비휘발성 기억 장소를 제공한다.

오퍼레이팅 시스템(235), 하나 이상의 애플리케이션 프로그램들(236), 다른 프로그램들(237) 및 프로그램 데이터(238) 를 포함해서 다수의 컴퓨터 프로그램들은 하드 디스크, 자기 디스크(229), 광 디스크(231), ROM(224) 또는 RAM(225) 에 기억될 수 있다.

사용자는 키보드(240) 및 포인팅 디바이스(242) (예를 들면, 마우스) 와 같은 다양한 입력 장치들을 통해 컴퓨팅 시스템(200) 에 커맨드들 및 정보를 입력할 수 있다. 실시간 데이터(256) 를 포착하거나 출력할 수 있는 카메라/마이크로폰(255) 또는 다른 유사 미디어 장치가 컴퓨팅 시스템(200) 의 입력 장치로서 포함될 수도 있다. 실시간 데이터(256) 는 적합한 인터페이스(257) 를 통해 컴퓨팅 시스템(200) 에 입력될 수 있다. 인터페이스(257) 는 시스템 버스(223) 에 연결될 수 있어서, 실시간 데이터(256) 가 RAM(225) 또는 다른 데이터 기억 장치들 중 하나에 기억되거나 처리될 수 있게 한다.

도시된 바와 같이, 모니터(247) 또는 다른 타입의 디스플레이 장치가 비디오 어댑터(248) 와 같은 인터페이스를 통해 시스템 버스(223) 에 연결된다. 모니터 외에, 컴퓨팅 시스템(200) 은 또한 스피커, 프린터 등과 같은 다른 주변 출력 장치들(도시되지 않음) 을 포함할 수도 있다.

컴퓨팅 시스템(200)은 원격 컴퓨터(249)와 같은 하나 이상의 원격 컴퓨터들의 논리 접속부들을 사용해서 네트워크 환경에서 동작할 수 있다. 원격 컴퓨터(249)는 다른 퍼스널 컴퓨터, 서버, 라우터, 네트워크 PC, 피어(peer) 디바이스, 또는 다른 흔한 네트워크 노드일 수 있고, 메모리 기억 장치(250)만이 도 2에 도시되어 있지만, 전형적으로 컴퓨팅 시스템(200)과 관련되어 상술된 소자들 다수 또는 모두를 포함한다.

도 2에 도시된 논리 접속부들은 근거리 통신망(local area network: LAN)(251) 및 광역 통신망(wide area network: WAN)(252)을 포함한다. 이러한 네트워크 환경들은 사무실에서 흔한 것으로, 기업간 컴퓨터 네트워크, 인트라넷, 및 인터넷이다.

LAN 네트워크 환경에서 사용될 때 컴퓨팅 시스템(200)은 네트워크 인터페이스 또는 어댑터(253)를 통해 근거리 통신망(251)에 연결된다. WAN 네트워크 환경에서 사용될 때 컴퓨팅 시스템(200)은 전형적으로 인터넷과 같은 광역 통신망(252)을 통한 통신을 설정하기 위해 모뎀(254) 또는 다른 수단을 포함한다. 내부 또는 외부 모뎀일 수 있는 모뎀(254)은 직렬 포트 인터페이스(246)를 통해 시스템 버스(223)에 연결된다.

네트워크 환경에서, 컴퓨팅 시스템(200)과 관련되어 기술된 컴퓨터 프로그램들 또는 그 일부는 원격 메모리 기억 장치에 기억될 수도 있다. 도시된 네트워크 접속부들은 예시적인 것으로 컴퓨터들 간의 통신 링크를 확립하기 위한 다른 수단이 사용될 수도 있음을 알 수 있다.

DVD2 API(108a- b)는 애플리케이션 저작(authoring)을 간단히 하고, 기능을 추가하며 DVD 플레이어 애플리케이션 개발에서 흔한 다수의 어려운 동기화 문제점들을 해결한다. 기본적으로, 일상적인 DVD API는 독립형 DVD 플레이어 애플리케이션으로서의 역할만을 행하는 디스커리지 소유권 주장 단일 사용 모놀리식 DVD 솔루션들(discourage proprietary single- use monolithic DVD solutions)을 돕는다. 이는 또한 다양한 애플리케이션들(예를 들어, 프리젠테이션 프로그램들, DVD 플레이어들, 게임들, 또는 대화식 학습 프로그램들)이 어떤 DVD 디코더 또는 DVD 하드웨어가 사용자 시스템을 지원하고 있는지를 인식할 필요 없이 DVD 지원을 추가하게 해 준다. 역사적으로, 커스텀(custom) DVD 솔루션들은 매우 하드웨어 의존적이고 사용자들을 위한 한정된 업그레이드 옵션들을 갖는 경향이 있다.

상세히 후술되는 바와 같이, DVD2 API(108a- b)는 DVD 네비게이터(106)에 대한 요청들의 완료 상태를 인식하는 애플리케이션을 위한 플렉시블 동기화 메카니즘들을 추가한다. 새로운 커맨드 완료 통지는 애플리케이션이 다른 작업들을 동시에 실행할 수 있게 해 주고 이전 요청의 상태를 알 수 있게 해 준다. 이전 DVD API들은 요청이 완료될 때까지 애플리케이션이 블럭되거나 또는 임의의 통지를 애플리케이션에게 송신하지 않는다고 생각했다. 이제 애플리케이션들은 대기를 위해 사용할 수 있는 동기화 객체를 수신하는 옵션을 갖거나 또는 완료 이벤트들에 대해 통지받는다.

동기화 메카니즘은 성공 여부를 나타내는 요청 상태를 리턴하거나 또는 실패의 이유(에러 코드)를 리턴한다. DVD 네비게이터(106)가 프로세싱을 실제로 개시했을 때 변경된 상태로 인해 후에 실패하게 되는 요청들을 이전 DVD API들이 성공적으로 실행한 것처럼 보이게 된다. 이 때에, 에러 표시를 플레이어 애플리케이션(102)에게 다시 전달할 방법이 없었다. 새로운 메카니즘은 디스크 프로그램(112)에 의해 또는 다른 사용자 액션에 의해 취소되거나 번복된 모든 요청을 플레이어 애플리케이션(102)에게 통지한다.

현재 DVD API들은 커맨드가 어떻게 현재 디스플레이와 상호 작용하는지를 나타내는 선정된 작용을 사용한다. 플레이어 애플리케이션이 새로운 요청을 발행할 때, 플레이될 임의의 콘텐트(비디오 또는 오디오)를 선취하고 취소한다. 대안으로, API 시맨틱스는 새로운 콘텐트가 제공되기 전에 현재 프리젠테이션이 완료될 것을 요구함으로써 사용자가 다른 액션을 요청하기 전에 대기하게 한다. DVD 플레이어들 및 게임들과 같은 상호 작용 애플리케이션들은 제1 작용(즉시 효과(instant effect))을 요구할 수도 있지만, 슬라이드쇼(slideshow)와 같은 다른 애플리케이션들은 제2 작용(

현재 프리젠테이션 완료)을 요구할 수도 있다. 이러한 2개의 옵션들이 상호 배타적이기 때문에, 선정된 API 시맨틱스가 둘 다 수용할 수는 없다. DVD2 API(108a- b)는 플레이어 애플리케이션(102)이 플래그들을 통해 원하는 행위(behavior)을 나타낼 수 있게 해 주고, 등기화 메카니즘과 어떻게 상호 작용하는지를 나타낼 수 있게 해 준다.

DVD 네비게이터(106)는 (한 세트의 메모리 레지스터들(124) (도 9 참조)의 형태로) 실행 상태(114)를 사용하는 가상 CPU를 시뮬레이트하도록 구성된다. 이전 DVD API들은 애플리케이션들이 레지스터들의 콘텐트들을 판독하게 해 준다. DVD2 API(108a- b)는 또한 플레이어 애플리케이션(102)이 메모리 레지스터들의 콘텐트들을 변경할 수 있게 해 준다. 결합된 판독/기록 기능은 도 9에 도시된 바와 같이 플레이어 애플리케이션(102)이 본질적으로 프로그램(112)과 " 통신" 할 수 있게 해 준다.

판독 및 기록 방법들은 등기화를 위해 사용될 수 있는 방식으로 작용한다. 예를 들어, 판독/기록 기능에 있어서, 플레이어 애플리케이션(102)은 모든 DVD 콘텐트(110) 또는 일부에 대한 " 제어 언로킹(controlled unlocking)" 또는 한정된 액세스를 구현할 수 있다. 제어 언로킹의 경우에, 사용자는 플레이어 애플리케이션(102)이 특정 메모리 레지스터들을 설정할 때까지 디스크의 뷰잉부로부터 한정될 수 있다. 플레이어 애플리케이션(102)은 콘텐트 저자, 사용자, 다른 프로그램, 웹사이트 등으로부터 상기 정보를 수신할 수 있다. 예를 들어, 도 12는 플레이어 애플리케이션(102)에 의해 레지스터들(124)에 기록되고 프로그램(112)에 의해 판독될 코드의 사용을 도시한다. 코드가 정확하면, DVD 콘텐트(110)의 일부(130)가 재생될 수 있다.

특정 구현들에서, DVD2 API(108a- b)는 DVD 명세서 부록 J에 제시된 가능성 있는 사용자 오퍼레이션들을 위한 간단한 명명 체계를 포함한다. DVD2 API는 보다 적은 DVD 전문어(jargon)를 사용하고 보다 직관적인 명명 체계를 특징으로 한다. DVD 명세서에 제안된 사용자 오퍼레이션 명칭들은 불명료하고 애플리케이션 프로그램들에 의한 용도 또는 언더- 유틸리제이션(under- utilization)을 부정확하게 할 수 있다. 명칭들은 추상 라벨 대신 용도를 제안한다. 또한, 시간 코드들은 불편한 BCD 코딩 대신 간단한 정수 포맷으로 리턴된다.

몇몇 이전 DVD API들은 DVD 네비게이터에게 분기가 항상 실패했음을 나타내는 에러 이벤트를 송신하게 함으로써 분기하는 최소 부모 레벨을 정확하게 처리하는 데 실패했다 (도 10 참조). 그 후 플레이어 애플리케이션은 부모 레벨을 증가시키고 영화를 처음부터 다시 시작해야만 했다. 분기가 실패하면, 플레이어 애플리케이션은 부모 레벨을 변경하도록 STOP 도메인을 입력하기 위해 재생을 정지할 필요가 있다. 영화를 다시 시작함으로써만 계속될 수 있다.

대조적으로, DVD2 API(108a- b)는 네비게이터(106)를 일시적으로 중단하고 네비게이터(106)가 계속되기 전에 플레이어 애플리케이션(102)이 부모 레벨 증가 요청에 응답하게 하는 모드를 갖는다. 증가 요청이 승인되면, 사용자가 영화를 처음부터 시작할 필요 없이 재생이 계속된다. DVD 명세서는 요청이 성공했는지 실패했는지를 인식할 때까지 네비게이터는 일시 중단해야만 한다고 기술한다. 이러한 작업을 달성하는 메카니즘을 기술하고 있지는 않으며 네비게이터가 " 플레이어에 내장된 임시 부모 레벨 변경 기능(Temporary Parental Level Change feature)을 호출" 할 것을 제안한다 (4.6.4.1 V14- 197).

DVD 명세서는 사용자가 멀티- 세그먼트 부모 레벨 분기들 (예를 들어, 도 11 참조)을 플레이하게 해주는 어떠한 메카니즘도 기술하고 있지 않다. 이와 같이, 이전 DVD API들은 현재 사용자 레벨에서 어떠한 분기도 허용되지 않은 경우 사용자가 멀티- 세그먼트 (또는 다중- 분기) 부모 레벨 분기들을 플레이하게 해주는 메카니즘을 제공하지 않았다. 과거에는, 현재 부모 레벨에 유용한 분기가 없었기 때문에, 네비게이터는 애플리케이션에게 재생이 정지되었다는 사실만을

통보했다.

대조적으로, 네비게이터(106) 및 DVD2 API(108a- b)는 재생 정지 통지와 함께 블럭을 플레이하고 값을 리턴하기 위해 필요한 최소 레벨을 계산한다. 애플리케이션은 그 후 DVD 콘텐트(110) 플레이를 계속하기 위해 필요한 부모 레벨을 사용자에게 통지할 수 있다. 따라서, 사용자는 더 이상 시도 및 에러를 통해 필요한 레벨을 추측할 필요가 없고, 따라서 각각의 시도 마다 영화를 다시 시작해야 할 필요가 없다.

게다가, DVD2 API(108a- b)는 DVD 부록 J 명세서 및 이전 DVD API들의 기능을 확장한다. DVD 부록 J 명세서는 실행할 액션들만을 규정한다. 플레이어 애플리케이션(102)이 디스크 또는 DVD 네비게이터의 상태(114)에 대한 정보를 어떻게 찾아내는 지에 대해서는 규정하지 않는다. 따라서, 새로운 디스크 및 네비게이션 상태 문의 기능이 제공된다.

이전 DVD API들과 달리, DVD2 API(108a- b)는 (예를 들어, API에 의해 리턴되는 데이터의 불완전한 기술로 인해) 애플리케이션 작성자(writer)에게 사용할 DVD 명세서의 복사본을 미리 준비할 것을 요구하지 않는다. 텍스쳐 정보, 제목 속성, 오디오 속성 및 서브픽쳐 속성을 얻기 위해 방법들에 의해 리턴되는 데이터는 애플리케이션 개발자들이 새로운 API 및 관련 문서로부터 필요한 정보를 획득할 수 있도록 문서화된다.

DVD2 API(108a- b)는 또한 애플리케이션이 단지 현재 제목 인덱스 대신 임의 제목 인덱스들의 속성들을 문의할 수 있게 해 준다. DVD2 API(108a- b)는 또한 지능적 음향 장치 애플리케이션들(intelligent Karaoke applications)이 구현될 수 있도록 오디오 스트림의 음향 장치 정보를 리턴한다. DVD2 API(108a- b)는 또한 애플리케이션이 사용자에게 구성 옵션들 (양방향 유사 프레임 스테핑, 매끄러운 되감기, 및 빨리 감기 등)을 제시하거나 또는 사용자 인터페이스를 지능적으로 변경할 수 있도록 디코더(116)의 기능을 리턴한다. 새로운 제어 기능이 또한 제공된다. 예를 들어, DVD2 API(108a- b)는 플레이어 애플리케이션(102)이 챕터 범위 또는 시간 범위들을 플레이해서 특정 메뉴 버튼들 (비관련 버튼들)을 선택하게 해 주고 사용자가 마우스 위치를 사용해서 버튼들을 선택하게 해 준다. 또한 북마크 객체들의 생성/설정을 지원하고 계산된 현재 고유 디스크 ID를 문의할 수 있도록 지원한다.

DVD2 API(108a- b) 및 관련 네비게이터(106)의 동기화 메카니즘을 더 잘 이해하기 위해, 애플리케이션에 대해 이하의 섹션들은 다양한 예의 오퍼레이션 모드들을 검사하고 몇몇 장점들 및 단점들을 지적한다. 본래, 다른 특정 변경물들과 함께 4개의 오퍼레이션 모드들이 있다. 초기 4개의 모드들은 도 3 내지 도 6에 도시되어 있다. 상기 모드들 각각은 본 발명에 따른 다양한 방법들 및 장치들에 의해 지원될 수 있다.

" don't care" 모드 또는 모델이 도 3에 도시되어 있다. 여기서, 플레이어 애플리케이션(102)은 결과가 있는 경우 결과가 무엇인지 및/또는 요청이 언제 완료되는지에 대해 신경쓸 필요 없이 네비게이터(106)에게 요청을 송신한다. 예를 들면, 위치 점프 요청, 쇼 메뉴 요청 등일 수 있다. 여기서, 플레이어 애플리케이션은 본래 요청된 오퍼레이션이 완료되었다고 생각한다.

도 4에 이벤트 모드 또는 모델이 도시되어 있다. 여기서, 플레이어 애플리케이션(102)은 네비게이터에 의해 송신된 포괄 이벤트 (요청이 완료된 때)에 대한 통보를 받는다. 이러한 모델의 단점은 플레이어 애플리케이션(102)이 하나 보다 많은 요청을 할 수 있고 이벤트들에게 개별적으로 알릴 수 없다는 점이다.

개선된 모델이 도 5에 도시되어 있다. 여기서, 이벤트가 플레이어 애플리케이션(102)에게 통보되는 대신, 네비게이터(106)는 요청 상태를 추적하기 위해 플레이어 애플리케이션(102)이 사용할 수 있는 객체를 생성한다. 이는 플레이어 애플리케이션(102)에게 인스턴스 추적을 행할 수 있는 능력을 제공한다.

도 6에 도시된 또 다른 개선된 모델에서, 네비게이터(106)는 추적에 사용될 수 있는 객체와 함께 다음 이벤트도 생성한다. 이러한 방식으로, 플레이어 애플리케이션(102)은 객체들을 사용해서 이벤트들을 개별적으로 알릴 수 있다. 따라서, 이러한 모델은 다중 인스턴스 추적(multiple instance tracking)을 지원한다.

다양한 모델들 및 DVD2 API(108a- b)에 대해 상세히 설명하기 전에, 블럭킹 전용 API 또는 논블럭킹 전용 API의 단점들이 기술될 것이다. 한 변경물이 도 7에 도시되어 있다. 여기서, 플레이어 애플리케이션(102)은 (물론, DVD2 API(108a- b)를 통해) 네비게이터(106)에게 요청을 송신한다. 플레이어 애플리케이션(102)은 네비게이터(106)로부터의 결과 메시지를 기다려야만 한다. 이러한 모델의 한가지 단점은 플레이어 애플리케이션(102)이 대기하는 동안 U/I(104)가 " 비동작 상태(frozen)" 가 될 가능성이 있다는 점이다.

비동작 U/I 문제점을 해결하는 한가지 방안은 도 8에 도시된 바와 같은 워커 프로그램(worker program)을 제공하는 것이다. 여기서, 워커 프로그램은 요청을 수신해서 네비게이터(106)에게 발송한 후 결과 메시지를 자체적으로 기다린다. 워커가 결과 메시지를 수신하면, 플레이어 애플리케이션(102)에게 발송한다. 이는 U/I(104)를 자유롭게 할 수는 있지만, 동시에 동작하는 몇몇 워커들을 관리하기가 어려울 수 있다.

대조적으로, 논블럭킹 API는 " don't care" 모드와 동등하다. 오퍼레이션 상태 또는 결과에 대한 직접적인 피드백은 없다. 애플리케이션은 재생 변경 (시간 변경, 메뉴 변경 등) 상태를 추측해야만 한다. 그러나, 디스크 콘텐트 및 구조 변경으로 인해, 이러한 방법은 매우 신뢰성이 없고 에러가 날 가능성이 있다. 이러한 관점에 따르면, 이하의 섹션들은 DVD2 API(108a- b) 사용에 대한 추가 세부 사항들을 제공한다.

이전 DVD API들의 IDVDControl 메소드들 모두는 애플리케이션에 대해 비동기식으로 작용한다 (논블럭킹(non- blocking) 모델). 따라서, 애플리케이션(102)이 메소드를 호출할 때, 네비게이터(106)는 예비된 검증을 실행하고 결과를 즉시 리턴한다. 그러나, 그동안, DVD 네비게이터의 상태가 변경될 수도 있고 DVD 네비게이터가 실제로 커맨드를 실행하기 시작할 때 요청이 실패할 수도 있다.

한 가지 해결점은 모든 요청들이 완료될 때까지 메소드들이 리턴하지 않음을 보장하도록 DVD API의 시맨틱스(semantics)를 변경하는 것이다. 비동기 동작을 유지하기 위해, 애플리케이션들은 개별적인 실행 경로들 (예컨대, 헬퍼 스레드들)을 생성해서 DVD API 호출들을 관리해야만 한다 (블럭킹 전용 모델에서 상술된 바와 같음). 멀티스레드 프로그래밍 모델들은 항상 애플리케이션 개발을 복잡하게 하고, 특히 간단한 스크립터블 인터페이스들을 복잡하게 한다.

따라서, 이러한 문제를 해결하기 위해, DVD2 API(108a- b)는 관련 동기화 커맨드 객체들을 생성한다. 커맨드 객체는 애플리케이션이 동기화하고 커맨드 상태에 대해 알도록 해 준다. 각각의 API 메소드는 2개의 여분의 인수들로 확장된다. DVD2 API 커맨드의 일반적인 형태는 다음과 같다:

HRESULT IDVDControl2::Command(arguments, dwFlags, IDvdCmd ** ppObj)

여기서: ppObject는 동기화 COM(Component Object Model) 객체를 애플리케이션(102)에게 리턴하는 데 사용되는 인수이고; dwFlags는 동기화 객체의 작용 및 용도를 결정하기 위해 메소드들에게 전달되는 플래그 집합이다. 유용한 선정된 플래그들의 비트 방식 집합(bit- wise union)이 있다.

동기화 객체는 이하의 인터페이스를 갖는다:

```
interface IDvdCmd : 1Unknown

{

HRESULT WaitForStart();

HRESULT WaitForEnd();

}
```

리턴된 객체는 애플리케이션에 의해 복구되어야만 한다. 미리 증가된 COM 객체를 리턴함으로써, 객체의 수명은 정확하게 유지될 수 있다. 인터페이스 변경은 애플리케이션이 시스템의 다른 변경과 함께 개시 및 종료 발생을 대기하게 해주는 2개의 메소드들을 포함함으로써 고유 인터페이스를 확장한다:

HANDLE GetStartHandle();

HANDLE GetEndHandle();

플래그들은 이하의 값들을 갖는다:

DVD_CMD_FLAG_SendEvents - 요청 상태에 대한 이벤트들이 송신됨.

DVD_CMD_FLAG_Block - 커맨드가 완료될 때까지 계속되지 않음.

DVD_CMD_FLAG_None - 플래그가 없음을 나타내는 위치홀더.

특정 리턴 코드 VFW_E_DVD_CMD_CANCELLED는 초기 DVD API 메소드에 의해, IDvdCmd::WaitForStart 또는 IDvdCmd::WaitForEnd에 의해 또는 커맨드가 선취되어 더 이상 유효하지 않음을 나타내는 이벤트와 함께 리턴된다.

커맨드 객체의 C++ 사용의 샘플 일례는 다음과 같다:

IDvdCmd* pObj;

HRESULT hres = IDvdControl2- > PalyTitle(15, DVD_CMD_FLAG_None, & p0bj);

// 대기하거나 통보하지 말라(don't wait or notify)

pObj- > Release();

상술된 바와 같이, 플레이어 애플리케이션(102)은 이하의 내용들 중 임의의 방법을 사용해서 커맨드의 개시 및 완료를 결정할 수 있다: 커맨드 객체를 직접 사용해서, 커맨드 객체들을 사용하지 않고, 커맨드 관련 이벤트들을 청취해서, 커맨드의 다수의 인스턴스들을 추적하는 것을 돕기 위해 이벤트들 및 객체들의 결합을 사용해서 결정할 수 있다.

객체 사용(Using Objects)

IDvdCmd 포인터를 커맨드에 전달함으로써, 네비게이터는 새로운 IDvdCmd 객체를 할당 및 리턴한다. 인터페이스 메소드 IDvdCmd::WaitForStart() 호출은 커맨드가 시작할 때까지 블럭되고 IDvdCmd::WaitForEnd()는 커맨드가 완료할 때까지 대기한다. 커맨드가 취소되었으면, 네비게이터는 VFW_E_COMMAND_CANCELLED를 리턴한다. 객체를 갖는 애플리케이션이 실행된 후에, COM 객체를 프리(free)로 해 주기 위해 Release()를 호출해야만 한다. DVD API에 전달된 NULL 포인터는 어떠한 커맨드 객체도 애플리케이션에 리턴되지 않아야 하며 커맨드 실행이 표준 비동기 모드로 계속되어야 함을 나타낸다.

다른 2개의 메소드들 GetStartHandle() 및 GetEndHandle()은 개시 또는 종료 이벤트들이 발생하기를 대기하는 동안 처리될 다른 요청들 (디스크 I/O, 사용자 인터페이스 변경, 세마포어 변경, 언블럭킹 스레드, 다른 프로세스들과의 통신 등)을 애플리케이션으로 하여금 기다리게 해주는 시스템 특정 동기화 객체를 리턴한다. 그 후 애플리케이션은 WaitForStart() 또는 WaitForEnd() 메소드들을 호출해서 결과를 검색한다. 마이크로소프트 윈도우즈 API의 일례는 다음과 같다:

handleStart = GetStartHandle()

Signaled = WaitForMultipleObjects(handleDiscIO, handleUserInter, ..., handleStart)

If signaled = handleStart

Result = DvdCmd- > WaitForStart()

객체를 사용하지 않음(Not Using Objects)

객체를 관리하는 대신, 애플리케이션은 널 객체 포인터를 갖는 DVD_CMD_FLAG_Block 플래그를 간단히 규정할 수 있다. 커맨드는 완료되거나 또는 취소될 때까지 리턴되지 않는다. API는 동기화 동작을 에뮬레이트한다. 예를 들어:

HRESULT hres = IDvdControl2- > PlayTilte(uTitle, DVD_CMD_FLAG_Block, 0);

는 의미상 다음과 동등하다:

IDvdCmd* pObj;

HRESULT hres = IDvdControl2- > PlayTilte(uTitle, DVD_CMD_FLAG_Block,

& pObj);

If(succeeded(hres)) {

Hres = pObj- > WaitToEnd();

pObj- > Release();

}

이벤트 사용(Using Events)

DVD_CMD_FLAG_SendEvents 플래그를 규정하여 네비게이터가 다음의 이벤트들을 발생시키게 한다:

{EC_DVD_CMD_START, lParam1, HRESULT}

{EC_DVD_CMD_END, lParam1, HRESULT}

애플리케이션이 하나의 커맨드만을 동기화할 필요가 있으면 (또는 커맨드 인스턴스들을 구별하지 않으면), 동기화 객체는 하나도 필요하지 않고 이벤트들만 필요하다. NULL 객체 포인터는 DVD API 메소드에 전달되고 이벤트와 함께 송신된 lParam1 값은 항상 0으로 설정된다.

이벤트 및 객체 사용(Using Events and Objects)

양 객체들 및 DVD_CMD_FLAG_SendEvents 플래그를 규정함으로써, 애플리케이션은 상이한 커맨드들을 추적할 수 있다. DVD2 API 호출은 애플리케이션이 차후 참조를 위해 사용할 수 있는 객체를 리턴한다. 이벤트 통지가 송신될 때,

DVD2 API는 애플리케이션이 IDvdCmd 객체에 다시 사상할 수 있는 각각의 이벤트에 대한 고유 식별자 (또는 '쿠키(cookie)') IParam1을 생성한다. 쿠기 방법은 애플리케이션이 이벤트를 손실하는 경우 메모리를 누설하지 않음을 보장하고 DVD 네비게이터가 객체의 유효성을 검증하게 한다.

DVD2 API 메소드 IDvdInfo2::GetCmdFromEvent(IParam1)는 쿠키를 커맨드 객체 포인터에 사상한다. 애플리케이션은 각각의 이벤트들의 프로세싱을 완료한 후에 리턴된 포인터에 대한 COM " Release" 메소드를 호출해야만 한다. (대개 END 이벤트를 수신한 후에) 애플리케이션이 메시지와 함께 완전히 완료되면, 저장된 글로벌 커맨드 포인터에 대한 " Release" 를 호출해야만 한다.

블럭킹/논블럭킹의 일례(Example of Blocking/Non- Blocking)

이하의 예시적인 일례들은 IDvdControl2 인터페이스를 사용해서 동기화가 어떻게 달성될 수 있는지를 보여준다:

편의상, 몇몇 일례들은 EC_DVD_CMD 이벤트들로부터의 IParam1 값을 IDvdCmd 객체에 사상하는 데 사용된 다음의 유틸리티 함수(function)와 관련된다:

```
IDvdCmd* GetDvdCmd(LONG_PTR IParam)

{

IDvdCmd* pCmd;

pIDvdInfo2- > GetCmdFromEvent(iParam, & pCmd);

return pCmd;

}
```

비동기화(비동기 모델)(No synchronization(Asynchronous model))

애플리케이션은 액션을 요청하는 메소드를 호출한다:

```
HRESULT hres = IDvdControl2- > PlayTilte(uTitle, 0, NULL);
```

이벤트들 없는 동기화(Synchronization without events)

이벤트를 사용하지 않고 커맨드가 종료하기를 대기하는 정확한 방법의 일례는 다음과 같다:

```
IDvdCmd* pObj;

HRESULT hres = IDvdControl2- > PlayTilte(uTitle, 0, & pObj);

If(SUCCEEDED(hres)) {

pObj- > WaitToEnd();

pObj- > Release();

}
```

이벤트들을 사용하는 부모 동기화(Parental synchronization using events)

IDvdCmd 객체를 관리하지 않고 단일 이벤트를 동기화 함:

```
HRESULT hres = IDvdControl2- > PlayTilte(uTitle, DVD_CMD_FLAG_SendEvents, NULL);
Function ProcesEvent(type, IParam1, IParam2)
{
switch(type)
{
case EC_DVD_CMD_END:
HRESULT hres = IParam2; //결과 코드가 IPram2에 있음
break;
}
}
```

이벤트들을 사용하는 전체 동기화(Full synchronization using events)

이벤트를 사용해서 커맨드를 대기하는 정확한 방법의 일례는 다음과 같다:

```
//글로벌 코드에서
IDvdCmd* pGlobalObj = 0;
//주의 사항: pGlobalObj는 이벤트가 발생하기 전에 네비게이터에 의해 할당
//되고; 그렇지 않으면 이벤트는 pGlobalObj가 초기화 되기 전에 포인트( *I)
//에서 이벤트가 발생할 수 있음
HRESULT hres = IDvdControl2- > PlayTilte(uTitle, DVD_CMD_FLAG_SendEvents, & pGlobalObj);
//(*I)
If(FAILED(hres)) {
pGlobalObj = NULL;
}
...
```

이벤트 프로세싱 함수에서:

```
Function ProcessEvent(type, lParam1, lParam2)

switch(type)

{

case EC_DVD_CMD_END:

IDvdCmd* pObj = GetDvdCmd(lParam1);

HRESULT hres = lParam2;

If(NULL != pObj) {

//이벤트에 의해 리턴된 객체가 PlayTilte에 의해 리턴된

//글로벌 포인터와 일치하면 처리함

If(pGlobalObj = obj) {

ProcessCmdEnd...

pGlobalObj- > Release();

pGlobalObj = NULL;

}

pObj- > Release();

}

break;
```

이벤트들을 사용하는 전체 동기화 및 개별 이벤트 루프 스레드(Full synchronization using events and a seperate event loop thread)

이벤트를 사용해서 커맨드를 대기하는 정확한 방법의 일례는 다음과 같다:

```
//글로벌 코드에서

IDvdCmd* pGlobalObj = 0;

{

LockCriticalSection

HRESULT hres = IDvdControl2- > PlayTilte(uTitle,
```

```
DVD_CMD_FLAG_SendEvents, & pGlobalObj);
If(FAILED(hres)) {
pGlobalObj = NULL;
}
UnlockCriticalSection
}
Function ProcessEvent(type, lParam1, lParam2)
switch(type)
{
case EC_DVD_CMD_COMPLETE:
case EC_DVD_CMD_CANCEL:
{
CautoLock(globalCritSect);
IDvdCmd* pObj = GetDvdCmd(lParam1);
HRESULT hres = lParam2;
If(NULL = pObj) {
If(pGlobalObj == obj) {
pGlobalObj- > Release();
pGlobalObj = NULL;
}
pObj- > Release();
}
break;
}
```

재생 상호 작용 제어 메카니즘 예(Exemplary Playback Interactivity Control Mechanism)

이전 DVD API 커맨드들은 콘텐트의 임의 변경에 대해 플레이어 애플리케이션(102)이 현재 콘텐트 프리젠테이션을 중지하기를 원해서 새로운 콘텐트로 전환한다고 가정했다. 개선된 DVD2 API 커맨드들은 이하의 플래그들로 커맨드 객체 메카니즘을 확장한다:

DVD_CMD_FLAG_Flush

DVD_CMD_FLAG_StartWhenRendered

DVD_CMD_FLAG_EndAfterRendered

여기서, .._Flush 플래그는 새로운 콘텐트의 디스플레이를 시작할 수 있도록(전과 같이) 현재 콘텐트의 프리젠테이션이 즉시 중단되어야만 함을 나타낸다. 플래그의 부재는 현재 콘텐트 프리젠테이션이 먼저 종료되어야만 함을 나타낸다. .._..Rendered 플래그들은 각각의 커맨드의 개시 및 종료의 시맨틱스를 변경한다. 디폴트에 의해, 일단 프로세스되었으면 커맨드는 개시하고 종료한다. 새로운 플래그들은 각각 콘텐트 변경 결과들이 프로세스되고 제공될 때 개시 및 종료가 발생함을 나타낸다.

디스크 통신 메카니즘 예(Exemplary Disc Communication Mechanism)

DVD2 API(108a- b)는 플레이어 애플리케이션이 DVD 네비게이터의 범용 레지스터들(GPRM들)을 판독할 수 있게 해줄 뿐만 아니라, 다음을 사용해서 GPRM들을 설정할 수도 있게 해 준다:

IDvdInfo2::GetAllGPRMs(WORD pwRegisterArray[16])

IDvdControl2::SetGPRM(ULONG ulindex, WORD wValue, DWORD dwFlags, IDvdCmd ** ppCmd)

결합된 판독/기록 기능은 DVD 애플리케이션들이 디스크의 프로그램과 " 통신(communicate)" 할 수 있게 해 주고 '제어 언로킹' 또는 콘텐트에 대한 한정된 액세스를 구현할 수 있다. 애플리케이션은 GetAllGPRMs을 사용해서 현재 상태를 판독할 수 있고 SetGPRM을 사용해서 특정 레지스터를 설정할 수 있다.

SetGPRM 메소드는 또한 애플리케이션 및 DVD 네비게이터의 가상 CPU를 동기화하는 데 사용될 수 있다. SetGPRM 메소드는 DVD 네비게이터가 사용자 커맨드들을 프로세스하도록 허용된 기간 중에만 실행된다 (프리젠테이션 및 정지 위상(the Presentation and Still phases), 3.3.6.1 V13- 28). 네비게이션 커맨드 실행은 원자로 간주된다. 이러한 위상들이 발생할 때까지 GPRM 설정은 연기된다. 애플리케이션은 좌표를 보장하기 위해 커맨드 및 이벤트 메카니즘을 사용할 수 있다. 커맨드 객체들의 이벤트 메카니즘은 이벤트 통지 (예를 들면, 도메인 변경 또는 시스템 레지스터 변경)에 의해 직렬화된다. 애플리케이션은 SetGPRM을 호출하고 커맨드 완료 이벤트가 수신될 때까지 대기한 후, DVD 네비게이터의 상태 변경 (도메인 변경일 수도 있음)을 나타내는 이벤트를 대기할 수 있다.

디스크- 애플리케이션 통신(disc to application communication)을 달성하는 한 가지 방법은 이하의 의사 코드로 나타내질 수 있다:

디스크는 데이터를 송신하고 응답을 기다린다:

(은- 디스크 네비게이션 커맨드를 사용해서) 디스크는 GPRM 값을 변경함

디스크는 상태를 변경함 (예를 들어, 도메인을 변경).

GPRM 변경을 대기하는 루프들 (애플리케이션에 의해 야기됨)

애플리케이션은 GPRM 데이터를 수신하고 응답한다:

상태 변경을 대기함 (예를 들면, 디스크의 도메인 변경 대기)

GPRM 값을 판독함

SetGPRM을 사용해서 GPRM 값을 설정함.

애플리케이션- 디스크 통신을 달성하는 한 가지 방법은 이하의 의사 코드로 나타내질 수 있다:

애플리케이션은 데이터를 송신하고 응답을 대기함:

애플리케이션은 SetGPRM을 사용해서 데이터를 설정함

애플리케이션은 계속하기 전에 도메인 변경을 기다림

디스크는 데이터를 수신하고 응답을 리턴한다:

디스크는 GPRM을 판독함

디스크는 상태를 변경함 (예를 들어, 도메인을 변경).

문의(정보) 인터페이스 예(Exemplary Query(Info) Interfaces)

DVD 명세서가 임의의 데이터 검색 메소드들을 제시하지 않더라도, DVD2 API들은 그러한 기능을 제공한다. 메소드들의 리스트는 다음과 같다:

GetAllGPRMs

GetAllSPRMs

GetAudioLanguage

GetCurrentAngle

GetCurrentAudio

GetCurrentButton

GetCurrentDomain

GetCurrentLocation

GetCurrentSubpicture

GetNumberOfChapters

GetPlayerParentalLevel

GetSubpictureLanguage

GetTotalTitleTime

GetTitleParentalLevels

GetCurrentUOPS

GetCurrentVolumeInfo(IDVD1::GetDVDVolumeInfo)

GetDVDDirectory(IDVD1::GetRoot)

GetAudioAttributes([in] ULONG ulStream, [out] DVD_AudioAttributes *pATR);

GetCurrentVideoAttributes([out] DVD_VideoAttributes *pATR):

GetVMGAttributes([out] DVD_MenuAttributes *pATR);

GetTitleAttributes(ULONG ulTitle, [out] DVD_MenuAttributes *pMenu,

[out] DVD_TitleAttributes *pTitle);

GetSubpictureAttributes([in] ULONG ulStream,

[out] DVD_SubpictureAttributes *pATR);

GetButtonAtPosition(POINT point, [out] ULONG *puButtonIndex);

GetButtonRect(ULONG ulButton, RECT *pRect):

GetDefaultAudioLanguage(LCID *pLanguage, DVD_AUDIO_LANG_EXT *pAudioExt):

GetDefaultMenuLanguage(LCID *pLanguage):

GetDefaultSubpictureLanguage(LCID *pLanguage,

DVD_SUBPICTURE_LANG_EXT *pSubpictureExtension):

GetDVDTextLanguageInfo(ULONG ulLangIndex, ULONG *pulNumOfStrings,

LCID*pwLangCode, DVD_TextCharSet *pbCharacterSet);

GetDVDTextNumberOfLanguages(ULONG *pulNumOfLangs);

GetDVDTextStringAsNative(ULONG ulLangIndex, ULONG ulStringIndex, BYTE *

pbBuffer, ULONG ulMaxBufferSize, ULONG *pulActualSize, enum

DVD_TextStringType *pTyp);

GetDVDTextStingAsUnicode(ULONG ulLangIndex, ULONG ulStringIndex,

WCHAR* pchBuffer, ULONG ulMaxBufferSize, ULONG * pActualSize,

DVD_TextStringType * pType);

GetCmdFromEvent(LONG_PTR dwID, IDvdCmd ** ppCmd);

GetDecoderCaps(DVD_DECODER_CAPS * pCaps);

GetDiscID(LPCWSTR pszwPath, ULONGLONG * pullUniqueID):

GetKaraokeAttributes([in] ULONG ulStream, DVD_KaraokeAttributres * pATR);

GetMenuLanguages(LCID * pLang, ULONG uMaxLang, ULONG * puActualLang);

IsAudioStreamEnabled(ULONG ulStreamNum, BOOL * pbEnabled);

IsSubpictureStreamEnabled(ULONG ulStreamNum, BOOL * pbEnabled);

제어 인터페이스 예(Exemplary Control Interfaces)

1) 기간 재생 인터페이스

챕터 범위들을 플레이하는 것 외에, DVD2 API는 다음을 사용해서 시간 기간 플레이를 허용한다:

PlayPeriodInTitleAutoStop(ULONG ulTitle, DVD_HMSF_TIMECODE * pStartTime,

DVD_HMSF_TIMECODE * pEndTime, DWORD dwFlags, IDvdCmd ** ppCmd)

이러한 메소드에 따르면, 애플리케이션들 (예를 들면, 비디오 편집 프로그램들 및 게임들) 은 콘텐트의 임의의 부분들을 정확하게 재생할 수 있다. 커맨드 객체 메카니즘과 결합되어, 슬라이드쇼 프리젠테이션, 비디오 게임 인터루드(interlude) 또는 키오스크(kiosk) 와 같은 임의의 애플리케이션이 단일 DVD2 API 커맨드를 사용해서 구현될 수 있다.

2) 디폴트 언어 인터페이스

SelectDefaultAudioLanguage(LCIDLanguage, DVD_AUDIO_LANG_EXT

audioExtension)

SelectDefaultSubpictureLanguage(LCIDLanguage, DVD_SUBPICTURE_LANG_EXT

subpictureExtension)

상기 메소드들은 (사용자로부터) 애플리케이션들이 DVD 재생을 위한 디폴트 언어 선택 사항들을 설정할 수 있게 해 준다.

3) 버튼 인덱스 선택

애플리케이션은 다음 메소드를 통해 메뉴 네비게이션을 자동화할 수 있다

SelectButton(ULONG ulButton)

4) 북마킹 API들

애플리케이션들은 전체 DVD 상태를 저장 및 복원할 수 있다 (북마크 부모 참조)

GetState(IDvdState ** pStateData)

SetState(IDvdState * pState, DWORD dwFlags, [out] IDvdCmd * ppCmd)

5) 기타

AcceptParentalLevelChange(BOOL bAccept) - 이하의 " 최소 부모 레벨 분기" 섹션을 참조

SetGPRM(ULONG ulindex, WORD wValue, DWORD dwFlags, [out] IDvdCmd ** ppCmd) -

SetOption(DVD_OPTION_FLAG flag, BOOL bEnable) - 확장 가능 옵션 설정 메카니즘

최소 부모 레벨 분기를 조정하기 위한 메카니즘(Mechanism for coordinating minimum parental level branching)

DVD 명세서 (섹션 4.6.4.1 pV14- 197) 에 따르면, DVD 네비게이터가 ' SetTmpPML ' 커맨드 (임시 부모 관리 레벨 설정) 와 마주칠 때, 현재 레벨을 임시적으로 상승시키기 위해 애플리케이션으로부터의 허가를 요청해야만 한다 (" 플레이어에 내장된 임시 부모 레벨 변경 기능 호출"). 부모 레벨 변경이 허용되면, 네비게이터는 부모 레벨을 상승시키고 콘텐트의 한정된 피스로 분기한다. 그렇지 않으면, 다음 커맨드로 계속된다.

이전 DVD API의 시맨틱스 하에서, DVD 네비게이터가 SetTmpPML 명령을 실행할 때, 애플리케이션에게 PARENTAL_LEVEL_TOO_LOW 이벤트만을 송신한다. 그 즉시 계속해서 부모 레벨 변경이 실패한 것처럼 다음 커맨드를 실행한다. 애플리케이션이 이벤트를 수신하고, 재생을 정지하고, 사용자 인터페이스를 디스플레이해서 부모 레벨을 변경한 후, 영화를 처음부터 다시 시작한다. DVD 명세서에 따르면, 네비게이터는 STOP 도메인 내에 있을 때만 부모 레벨을 변경할 수 있다. 그 결과, 네비게이터가 변경시 일시 중지하지 않기 때문에, 재생을 정지해야만 한다.

예를 들어, DVD2 API(108a- b) 의 경우, 이하의 시퀀스가 발생할 수 있다. 애플리케이션은 메소드를 호출함으로써 부모 레벨 변경 기능의 유용성을 API에게 통지한다:

IDVDControl2::SetOption(DVD_NotifyParentalLevelChange, TRUE)

DVD 네비게이터가 SetTmpPML 명령과 마주칠 때, 애플리케이션에게 PARENTAL_LEVEL_TOO_LOW 이벤트를 송신한다. 애플리케이션은 사용자가 부모 레벨을 증가시킬 수 있도록 몇몇 사용자 인터페이스를 디스플레이할 것으로 기대된다. DVD 네비게이터는 TRUE 또는 FALSE를 갖는 IDVDControl2::AcceptParentalLevelChange() 를 호출함으로써 애플리케이션이 응답할 때까지 블럭된 후 재생을 정지시킬 필요 없이 진행된다.

멀티- 세그먼트 부모 레벨 분기 재생을 돕는 메카니즘(Mechanism for aiding playback of multi- segment parental level branches)

DVD 명세서 (섹션 4.1.4.1 V14- 22) 는 현재 부모 레벨을 기초로 상이한 프로그램 체인들 (대개 콘텐트의 상이한 가능한 세그먼트들) 을 선택하기 위한 체계에 대해 기술하고 있다. 예를 들어, 비디오의 특정 지점에서, 한 장면의 상이한 버전들이 가능한데, 부모 레벨 (예를 들면, PG, R 정격용으로 의도된 세그먼트들 또는 자식) 을 기초로 네비게이터에 의해 자동으로 선택된다.

각각의 제목에 있어서, PTL_MAI 테이블은 현재 부모 레벨을 16- 비트 마스크에 사상한다. 재생 중에, DVD 네비게이터는 PTL_MAI 테이블로부터 현재 부모 비트 마스크를 획득한다. 부모 비트 마스크는 네비게이터가 부모 블럭(각각의 프로그램 체인이 배타적 부모 비트 마스크를 가지는 프로그램 체인들의 집합)과 마주칠 때 사용된다. 네비게이터는 현재 부모 비트 마스크와 공통 비트들을 공유하는 비트 마스크를 갖는 프로그램 체인을 위해 VTS_PGCI_SRP (섹션 4. 2.3 V14- 62)의 각각의 PTLID_FLD를 탐색한다.

프로그램 체인이 부분적으로 현재 비트 마스크와 일치하지 않으면, DVD 네비게이터의 이전 버전들이 재생을 중지하고 DVD_ERROR_LowParentalLevel 이벤트를 애플리케이션에게 송신한다.

사용자를 돕기 위해, DVD2 API(108a- b)의 특정한 구현예들은 사용자가 계속할 수 있게 해주는 최소 필요 부모 레벨을 계산하기 위해 이하의 알고리즘을 사용한다:

PTL_MASK= 0으로 초기화 한다 (가능한 허용 부모 레벨들)

VTS_PGCI_SRP의 각각의 프로그램 체인 인덱스 i인 경우

IfVTS_PGCI_SRP[i].BlockType= 1 (부모 블럭에서)

PTL_MASK= PTL_MASKunionVTS_PGCI_SRP[i].PTL_ID_FLD

IfPTL_MASK= 0 then

부모 레벨이 존재하지 않아서, 임의의 레벨이 작용함

Else

PTL_MAI의 각각의 부모 레벨 인덱스 i인 경우

LetPTL_LVLI= PTL_MAI[8- i]

IfPTL_LVLI[title_index] & PTL_MAI[8- i]= 0

(주의 사항: VMGM 도메인에서title_index= 0)

Returni

인덱스 i는 DVD_ERROR_LowParentalLevel 이벤트와 함께 리턴된다. 애플리케이션(102)은 사용자에게 설정된 가능한 부모 레벨을 제안하기 위해 인덱스를 사용할 수 있다.

북마킹(Bookmarking)

DVD 네비게이터(106)는 플레이어 애플리케이션(102)이 DVD 재생의 현재 상태(114)를 인코드해서 데이터의 영구 블럭을 포함하는 추상 객체 (북마크(150)라고 함)로 저장할 수 있도록 구성된다. 도 13은 북마킹 기능의 일례를 도시한다.

용도를 더 추상화하고 간단히 하기 위해, DVD2 API(108a- b)는 북마크에 포함된 상태 정보를 저장, 복원 및 문의하도록 구성된다. 플레이어 애플리케이션(102)은 네비게이터(106)를 사용해서 북마크(150)의 정보를 문의하고 차후

사용을 위해 저장할 수 있다. 플레이어 애플리케이션(102)은 DVD 네비게이터(106)에게 북마크에 포함된 DVD 재생 상태(114)를 복원하도록 명령함으로써 후에 재생을 재개할 수 있다. 북마크들을 복원함으로써 플레이어 애플리케이션은 DVD 콘텐트(110)의 임의의 위치로부터 또한 임의의 번호로부터 플레이를 시작할 수 있다. 북마크들은 단기(메모리) 기억 장치 또는 장기 기억 장치(예를 들면, 하드 드라이브)에 저장될 수 있고, 플레이어 애플리케이션(102) 및/또는 PC가 셧다운된 후 다시 시작된 후에도 복원될 수 있다. 북마크는 DVD 네비게이터의 상태(예를 들면, 내부 레지스터 값, 재생 위치, 재생 상태)를 포함할 뿐만 아니라, 플레이 중인 현재 디스크 콘텐트 및 사용자 설정에 관한 정보도 포함한다. 플레이어 애플리케이션(102)은 예를 들어 특정 디스크(대체로, 플레이될 디스크)를 위해 플레이될 수 있는 사전 저장 북마크들 중에서 적합한 북마크를 지능적으로 선택하기 위해 여분의 정보를 사용할 수 있다. 북마크들은 사용자들 간에 또한 다양한 애플리케이션들 간에 공유될 수 있다.

북마크 추상 데이터 타입은 2개의 양상들로 이루어진다; 1) 실제 북마크(150) 자체, 및 2) 북마크에 포함된 정보를 저장, 복원, 및 문의하는 데 사용되는 API 호출들. 특정한 구현예들에 따르면, 북마크(150)는 적어도 다음과 같은 정보를 포함한다: 실제로 고유한 디스크 식별자(145), 디스플레이 중인 현재 비디오 객체 유닛(video object unit: VOBU)의 어드레스(DVD 명세서의 섹션 5.1.1), 루프 카운트 및 셔플(shuffle) 히스토리(DVD 명세서의 섹션 3.3.3.2), 현재 DVD 재개 정보(DVD 명세서의 섹션 3.3.3.2에 약술됨), 현재 DVD 범용 파라미터(GPRM) 및 시스템 파라미터(SPRM) 값들(섹션 4.6.1.1 및 4.6.1.2), 및 현재 도메인 및 위상(섹션 3.3.3 및 3.3.6). 다른 특정 구현들에서, 북마크는 또한 버전(versioning) 및 무결성(integrity) 정보를 포함한다. 북마크(150)는 저장을 위해 추상 객체로서 또는 2진 데이터 블럭으로서 팩키지화될 수 있다.

이러한 북마크 기술을 제공하기 위해, 특정한 구현예들의 DVD2 API(108)는 다음과 같은 메소드들을 지원한다:

1. 현재 위치로부터 북마크를 생성함.

Bookmark = GetBookmark()

2. DVD 네비게이터가 위치를 북마크로 변경시키게 함.

SetBookmark(bookmark)

3. 북마크가 가리키는 디스크를 찾기 위함.

DiscID = GetDiscIdentifierFromBookmark(bookmark)

4. 북마크의 2진 표현으로의 변환 및 2진 표현의 북마크로의 변환

BinaryData(data,size) = ConvertBookmarkToBinary(bookmark)

Bookmark = ConvertBinaryToBookmark(BinaryData)

현재 위치 저장 또는 전력 절약 기능(예를 들면, 복원될 수 있는 저전력 상태로 들어가기 위해 컴퓨터 상태를 저장할 수 있는 기능)을 구현하기 위한 애플리케이션 의사코드:

Bookmark = GetBookmark()

BinaryData(data,size) = ConvertBookmarkToBinary(bookmark)

BinaryData(data,size)를 저장함

셧다운 또는 전력 절약 모드로 들어감

전력 절약 모드로부터 리턴할 때, DVD 재생을 재개하기 위해서는 다음과 같이 하라:

BinaryData(data,size) 를 불러 옴

Bookmark = ConvertBinaryToBookmark(BinaryData)

If GetDiscIdentifierFromBookmark(bookmark) = 현재 디스크 Id

Then

SetBookmark(bookmark)

지능적 북마크들을 구현하기 위한 애플리케이션 의사코드의 일례

각각의 저장 북마크 " bookmark" 의 경우

If GetDiscIdentifierFromBookmark(bookmark) = 현재 디스크 Id

Then

북마크를 사용자 선택 가능 리스트에 추가하라

고유 식별자 생성(Unique Idetifier Generation)

현재 DVD 명세서는 각각의 디스크에 대한 내장 고유 식별자 (" DVD 고유 식별자") 를 갖는다. 그러나, 애플리케이션들은 디스크 저자가 식별자를 정확하게 구현했다고 가정해야만 한다; 불행히도, 항상 그렇지는 않다.

다수의 애플리케이션들은 DVD 디스크를 식별하기 위해 고유 태그를 필요로 한다. 예를 들어, 사용자가 DVD 디스크들을 교환할 때, 재생 시스템은 새로운 디스크를 갖고 있는지를 결정할 필요가 있다. 새로운 디스크를 갖고 있으면, 재생을 리셋해야만 하고, 그렇지 않으면, 사용자의 뷰잉을 방해하지 않고 계속할 수 있다. 디스크들을 구별할 수 없으면, 항상 리셋해야만 한다. 고유 식별자(145) (도 13 참조) 는 상이한 디스크들을 구별할 수 있게 해 준다 (그러나, 상이한 정확한 복사본들은 구별하지 못함) .

고유 식별자(145) 는 또한 애플리케이션들이 특정 DVD 디스크와의 저장된 정보의 호환성을 검증하게 해 준다. 애플리케이션들은 잘못된 디스크를 갖는 캐시 정보를 성공적으로 사용할 수 없다. 예를 들어, 사용자가 북마크를 사용해서 디스크의 저장된 위치를 다시 호출하고자 할 때, DVD 네비게이터(108) 는 북마크에 저장된 고유 식별자를 현재 디스크의 고유 식별자와 비교함으로써 데이터의 호환성을 보장할 수 있다. 식별자들이 일치하는 경우에만 재생이 계속된다.

고유 식별자들(145) 은 애플리케이션들이 데이터베이스에 대한 인덱스로서 고유 식별자를 사용해서 디스크와 추가 정보를 연관시킬 수 있게 해 준다. 예를 들어, DVD 명세서가 디스크에 대한 텍스쳐 정보를 지원하더라도, 거의 사용되지 않는다. 디스크의 제목 및 콘텐트의 웹- 베이스 데이터베이스는 디스크의 식별자를 계산한 후에 애플리케이션들에 의해 저장 및 검색될 수 있다.

DVD 디스크의 현재 내장 고유 식별자는 부적절하다. 첫째로, 식별자는 크기가 너무 크고 (32 바이트) , 실제로 고유한지를 보장하기 위해 디스크 저자에 의존하며, 중앙 엔티티(central entity) 가 회사들 간의 고유성이 유지되는지를 보장하기 위해 디스크 저자들에 대한 식별자들의 범위들을 할당해야만 한다.

다른 종래의 " 고유" 식별자 알고리즘들은 다수의 디스크들을 위한 고유 식별자들을 생성하지 않는다. 여기서, DVD 디스크들의 총 수가 증가함에 따라 2개의 디스크들이 동일한 식별자를 할당받을 확률은 지수 함수적으로 증가한다. DVD 디스크들의 예상 증가 추세에 따라, 다수의 '고유' 식별자 루틴들은 부적절하게 된다. 또한, 상기 알고리즘들은 종종 공지되고 및/또는 증명할 수 있는 특성들을 갖지 않는다. 공지된 특성들 없이, 생성된 식별자들의 효율성(effectiveness) 또는 적합성(suitability)을 말할 수 없다.

본 발명의 특정 구현예들에 따르면, 고유 식별자(145)는 DVD의 VIDEO_TS 디렉토리의 다양한 파일들의 파일 헤더 및 파일 콘텐트의 연접 2진 표현 또는 배열된 2진 표현의 64- 비트 CRC를 계산함으로써 생성된다. 이러한 가능성은 도 13 및 도 14에 도시되어 있다.

UniqueID2 알고리즘은 4개의 단계들로 식별자를 생성한다:

단계 1. VIDEO_TS 디렉토리의 파일명들은 수집되어 알파벳 순으로 정렬된다.

단계 2. 각각의 파일로부터의 파일 헤더들이 CRC로 계산된다.

단계 3. VMGI 파일(" VIDEO_TS \ VIDEO_TS.IFO")로부터의 데이터는 CRC로 계산된다.

단계 4. VTSI 파일(" VIDEO_TS \ VTS_xx_O.IFO")로부터의 데이터는 CRC로 계산된다.

64- 비트 CRC는 필드 GF(2)의 양분할 수 없는(irreducible) 다항식을 사용해서 계산된다. 다항식의 일례는 다음과 같다:

$$P_{64} = x^{64} + x^{61} + x^{58} + x^{56} + x^{55} + x^{51} + x^{50} + x^{47} + x^{42} + x^{39} + x^{38} + x^{35} + x^{33} + x^{32} + x^{31} + x^{29} + x^{26} + x^{25} + x^{22} + x^{17} + x^{14} + x^{13} + x^{9} + x^{8} + x^{6} + x^{3} + x^{0}$$

다항식은 x" - 1이 64도(degree)의 양분할 수 없는 (소수(prime)) 인수를 갖도록 지수 n을 찾음으로써 생성된다.

실제 CRC 값은 특정예들에서 모든 2진 데이터를 단일 블럭 (비트들 b_0 내지 b_n)으로 연접(concatenating)하고, 각각의 비트 b_i를 다항식의 계수 x^i에 할당한 후, 다항식 P_{64}로 나눈 후의 나머지를 계산함으로써 계산된다:

$$CRC_{64} = \left[\sum_{i=0}^{n} b_i x^i \right] \bmod p_{64}$$

예시가 되는 구현은 다음과 같다:

단계 1

VIDEO_TS 디렉토리의 파일명들은 수집되어 리스트에 알파벳 순서로 정렬된다.

단계 2

리스트의 각 파일명에 있어서, 이하의 구조가 채워지고 CRC에 추가된다 (모든 데이터 필드들은 이전 LSB에 있음):

부호 없는 64 비트 정수: dateTime (1601년 1월 1일로부터 100 나노세컨드 간격들로 경과된 시간)

부호 없는 32 비트 정수: dwFileSize

BYTE bFilename[filename Length]

BYTE bFilenameTermNull = 0

단계 3

존재하면, 파일 " VIDEO_TS \ VIDEO_TS.IFO" 의 처음 65536 바이트들이 판독되고 CRC에 추가된다. IFO 파일이 65536보다 작으면, 전체 파일이 추가된다.

단계 4

존재하면, 파일 " VIDEO_TS \ VTS_01_O.IFO" 의 처음 65536 바이트들이 판독되고 CRC에 추가된다. IFO 파일이 65536보다 작으면, 전체 파일이 추가된다.

본 발명의 다양한 방법들 및 장치들의 몇몇 양호한 구현들이 첨부된 도면들에 도시되었고 상술된 상세한 설명에 기술되었지만, 본 발명은 기술된 구현예들로만 제한되지 않고, 이하의 청구항들에서 설명되고 정의된 본 발명의 범위 내에서 다양하게 재구성, 변경, 및 대체될 수 있음을 알 것이다. 또한, 상술된 참조 문헌들 각각은 참조용으로 전체가 명백하게 본 명세서에 인용되어 있다.

발명의 효과

본 발명의 특성들에 따르면, 포괄 네비게이터 컴포넌트의 성능을 더 확장시키기 위해 확장 소자들이 개발되었다. 포괄 네비게이터 프로그램과 함께 사용될 향상된 API들이 제공된다. 플레이어 애플리케이션과 양호하게 동기화하도록 네비게이터 프로그램이 확장되고, 그렇게 함으로써 플레이어 애플리케이션을 위한 양호한 제어 메카니즘들, 한정/부모 제어 콘텐트 뷰의 향상된 시행, 콘텐트 소스에 대한 고유 식별자들의 생성 및 할당과, 재생 시작 및 정지를 위한 향상된 사용자 및 플레이어 애플리케이션 환경이 제공된다.

(57) 청구의 범위

청구항 1.

메모리; 및

상기 메모리에 동작적으로(operatively) 결합되고 매체로부터의 멀티미디어 콘텐트를 액세스하도록 동작적으로 구성 가능한 논리부(logic)

를 포함하되,

상기 논리부는 멀티미디어 네비게이터 프로그램, 제어 애플리케이션 프로그래밍 인터페이스(API), 및 정보 API를 제공하고, 상기 제어 API 및 상기 정보 API는 적어도 하나의 오퍼레이션이 실행될 지를 선택적으로 결정하는 플래그들에 응답하도록 구성되고, 상기 오퍼레이션은 플레이어- 네비게이터 동기화 오퍼레이션, 선택적 상호 작용 사용자 오퍼레이션, 및 판독/기록 레지스터 오퍼레이션을 포함하는 오퍼레이션 그룹으로부터 선택되는 장치.

청구항 2.

제1항에 있어서,

상기 플레이어- 네비게이터 동기화 오퍼레이션은

멀티미디어 플레이어 애플리케이션이 상기 네비게이터 프로그램에 요청 커맨드를 출력하게 하고; 그 후에 멀티미디어 콘텐트 네비게이터 프로그램이 상기 요청된 커맨드의 시작, 완료, 또는 취소에 따라 상기 플레이어 애플리케이션에게 이벤트 식별자 및 상태 결과를 리턴하게 하는 동기화 단계들을 실행하는 장치.

청구항 3.

제2항에 있어서,

상기 요청 커맨드 및 상기 이벤트 식별자 모두는 상기 네비게이터 프로그램과 동작적으로 관련된 적어도 하나의 애플리케이션 프로그래밍 인터페이스(API)를 통해 전달되는 장치.

청구항 4.

제3항에 있어서,

상기 API는 상기 요청된 커맨드의 시작, 완료, 또는 취소에 따라 상기 이벤트 식별자 및 상태 결과가 리턴되어야만 함을 식별하기 위해 상기 플레이어 애플리케이션에 의해 선택적으로 설정된 적어도 하나의 플래그 값에 응답하도록 더 구성되는 장치.

청구항 5.

제2항에 있어서,

상기 네비게이터 프로그램은 상기 요청 커맨드에 응답해서 멀티미디어 정보를 동작적으로 액세스하도록 구성되는 장치.

청구항 6.

제5항에 있어서,

상기 멀티미디어 정보는 DVD(digital versatile disc) 포맷 콘텐트를 포함하는 장치.

청구항 7.

제1항에 있어서,

상기 메모리는 적어도 하나의 레지스터를 제공하고, 상기 선택적 상호 작용 사용자 오퍼레이션은 멀티미디어 플레이어 애플리케이션이 멀티미디어 네비게이터 프로그램과 동작적으로 연관된 적어도 하나의 레지스터에 데이터를 기록하게 하고 상기 멀티미디어 콘텐트 내에 정의된 적어도 하나의 프로그램이 상기 적어도 하나의 레지스터를 판독할 수 있게 하는 장치.

청구항 8.

제7항에 있어서,

상기 데이터는 코드를 포함하고, 상기 적어도 하나의 프로그램은 나머지 멀티미디어 콘텐트의 적어도 일부분이 액세스되게 함으로써 상기 코드에 응답하는 장치.

청구항 9.

제7항에 있어서,

상기 멀티미디어 콘텐트는 DVD(digital versatile disc) 포맷 콘텐트를 포함하는 장치.

청구항 10.

제9항에 있어서,

상기 데이터는 상기 DVD 포맷 콘텐트와 연관된 정확한 재생 정보를 포함하는 장치.

청구항 11.

제10항에 있어서,

상기 정확한 재생 정보는 제목, 개시 시간 및 종료 시간을 포함하는 장치.

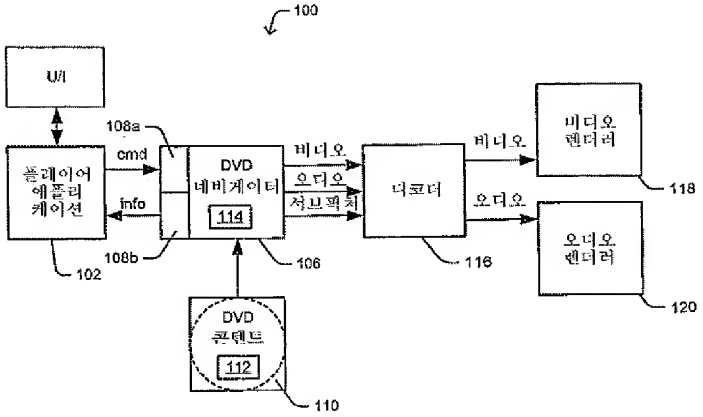
청구항 12.

제8항에 있어서,

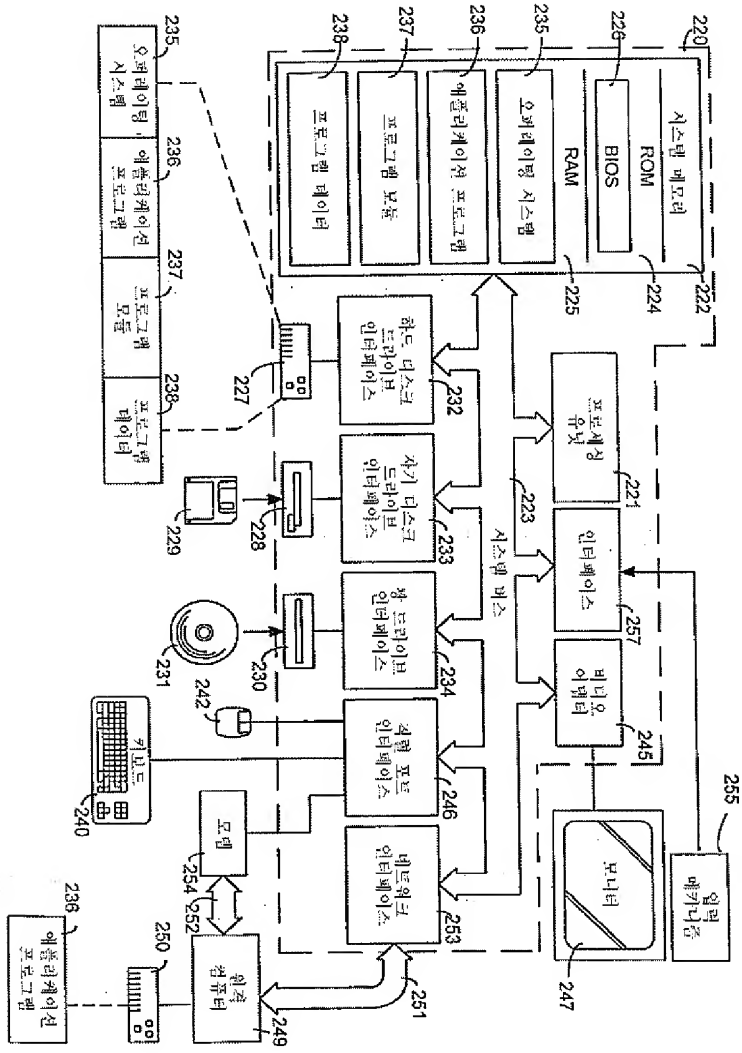
상기 멀티미디어 플레이어 애플리케이션은 상기 멀티미디어 네비게이터 프로그램과 동작적으로 연관된 적어도 하나의 애플리케이션 프로그래밍 인터페이스(API)를 통해 상기 적어도 하나의 레지스터에 데이터를 기록하는 장치.

도면

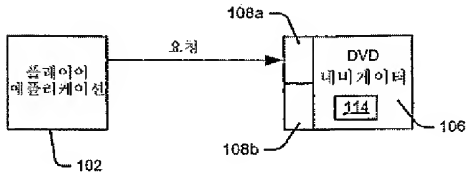
도면 1



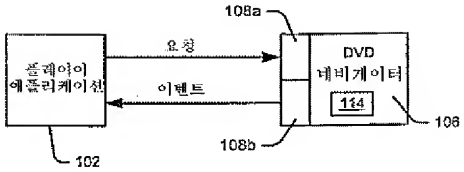
도면 2


220
시스템 메모리
ROM
BIOS
RAM
226
224
222
오퍼레이팅 시스템
235
에플리케이션 프로그램
236
프로그램 모듈
237
프로그램 데이터
238
225
하드 디스크 드라이브 인터페이스
232
자기 디스크 드라이브 인터페이스
233
광 드라이브 인터페이스
234
직렬 포트 인터페이스
246
네트워크 인터페이스
253
모뎀
254
252
프로세싱 유닛
221
223
시스템 버스
인터페이스
257
비디오 어댑터
245
모니터
247
255
오디오 캡처 메커니즘
227
229
228
231
230
242
키보드
240
250
원격 컴퓨터
249
251
236
에플리케이션 프로그램

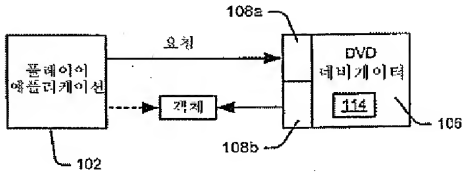
도면 3


108a
요청
호환 에플리케이션
102
DVD 네비게이터
114
106
108b

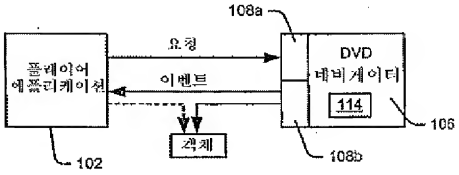
도면 4



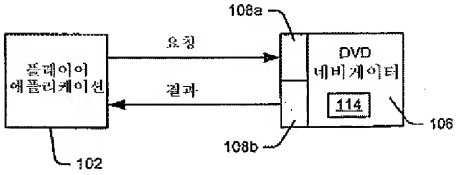
도면 5



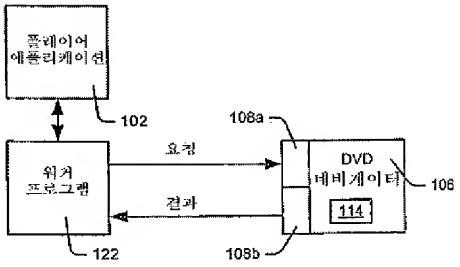
도면 6



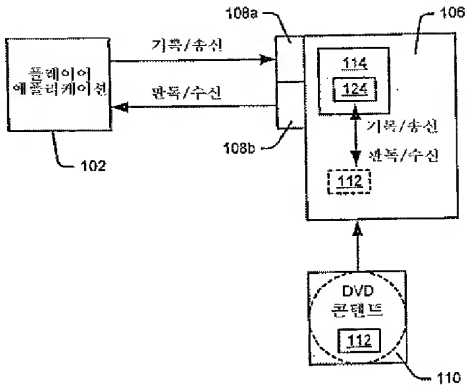
도면 7



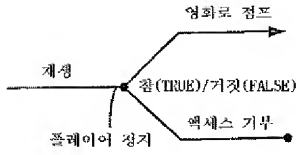
도면 8



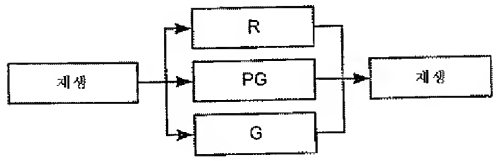
도면 9



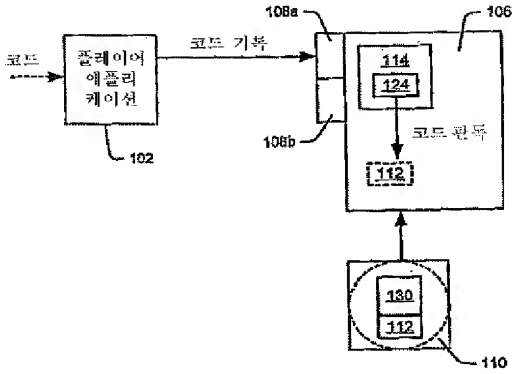
도면 10



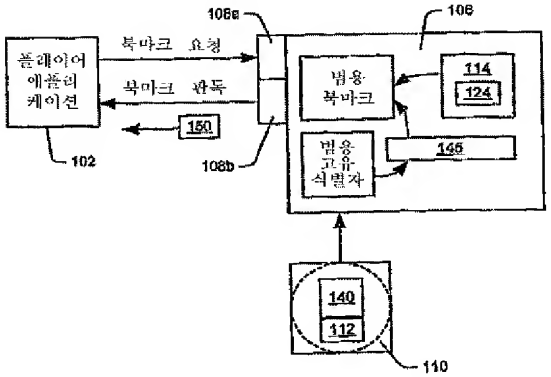
도면 11



도면 12



도면 13



도면 14